An Integrated Package of Neuromusculoskeletal Modeling Tools in Simulink™

R. Davoodi, I.E. Brown, N. Lan, M. Mileusnic and G.E. Loeb
A.E. Mann Institute for Biomedical Engineering, University of Southern California
Los Angeles, CA 90089, USA, http://ami.usc.edu

Abstract – **An integrated neuromusculoskeletal (NMS) modeling tool has been developed to facilitate the study of the control of movement in humans and animals. Blocks representing the skeletal linkage, sensors, muscles, and neural controllers are developed using separate software tools and integrated in the powerful simulation environment of Simulink (Mathworks Inc., USA). Musculoskeletal Modeling in Simulink (MMS) converts anatomically accurate musculoskeletal models generated by SIMM (Musculographics Inc., USA) into Simulink blocks. It also removes runtime constraints in SIMM, and allows the development of complex musculoskeletal models without writing a line of code. Virtual Muscle builds realistic Simulink models of muscle force production under physiologic and pathologic conditions. A generic muscle spindle model has also been developed to simulate the sensory output of the primary and secondary afferents. Neural control models developed by various Matlab (Mathworks Inc., USA) toolboxes can be integrated easily with these model components to build complete NMS models in an integrated environment.**
***Keywords* - Musculoskeletal modeling, Simulink**

I. Introduction

Movement in humans and other animals is the result of complicated interactions involving voluntary command signals, sensory receptors, reflex circuits, muscle actuators, skeletal linkages, and external forces. In order to understand the integrated system or to replace damaged portions with neural prosthetic components, the system must be studied in its entirety. Most of our current knowledge about sensorimotor control is the result of direct measurements from the neurons, muscles, and limbs of naturally moving subjects, which are difficult and limited.

Computer models of NMS systems can extend and complement experimental studies. Model parameters are accessible for inspection and/or modification. The performance of the system can be simulated with different versions of the components in order to understand pathological conditions and to develop strategies for their treatment. The validity of such predictions, however, depends on the accuracy and completeness of the models. Developing NMS models is often as challenging as recording data from behaving subjects. Such models must then be integrated with models of the biological and prosthetic neural components.

Current modeling tools in the market are inadequate for development of a complete NMS model. SIMM is the only commercial software for developing anatomically realistic musculoskeletal models [1]. It also provides a Dynamic Pipeline™ tool to SD-FAST, a commercial package for generating and solving equations of motion. The SIMM user must be familiar with the structure of the SIMM-generated C-programs and must write additional C-programs for model components such as sensors, command signals, and controllers. SIMM also has substantial limitations on its ability to incorporate run-time changes of muscle excitation and external forces, which handicap its use to study control algorithms [2]. Furthermore, the models of muscle force generation in SIMM are relatively primitive and do not represent more complex dynamical or pathological states.

To overcome these limitations, many researchers have expended great effort to develop their own models for specific applications. Because these models are usually developed in various programming and simulation environments most familiar to the developer, they are often difficult to share or reuse in other modeling studies. Therefore, there is a clear need for an integrated modeling environment which:

- Provides the basic NMS model components
- Facilitates sharing and reuse of the model components
- Does not require sophisticated programming skills
- Is powerful and accurate
- Is easy to learn and use

We chose Simulink as an integrated modeling environment that is already in widespread use for building models of complex systems. Simulink provides a graphical interface to help the user assemble and navigate multi-component systems. The block-oriented structure of Simulink facilitates reuse and sharing of code among researchers. Because Simulink runs on top of Matlab (Mathworks Inc., USA), the user can invoke the powerful Matlab language and toolboxes to build components, organize simulation sequences and graphically render the results.

For easier and faster development of complete NMS models in Simulink, we have developed software tools to build Simulink models of typical NMS model components.

II. Methods

The main components of a typical NMS model are the sensory receptors, neural controllers in the central nervous system (CNS), muscles actuators, skeletal system, and the interactions with the environment. We have developed several software utilities to enhance the Simulink capabilities for easier development of the complete NMS models.

Report Documentation Page

Report Date	Report Type	Dates Covered (from... to)
25 Oct 2001	N/A	-

Title and Subtitle An Integrated Package of Neuromusculoskeletal Modeling Tools in Simulink	**Contract Number**
	Grant Number
	Program Element Number
Author(s)	**Project Number**
	Task Number
	Work Unit Number
Performing Organization Name(s) and Address(es) A.E. Mann Institute for Biomedical Engineering University of Southern California Los Angeles, CA 90089	**Performing Organization Report Number**
Sponsoring/Monitoring Agency Name(s) and Address(es) US Army Research, Development & Standardization Group (UK) PSC 802 Box 15 FPO AE 09499-1500	**Sponsor/Monitor's Acronym(s)**
	Sponsor/Monitor's Report Number(s)

Distribution/Availability Statement
Approved for public release, distribution unlimited

Supplementary Notes
Papers from 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, October 25-28, 2001, held in Istanbul Turkey. See also ADM001351 for entire Conference on cd-rom., The original document contains color images.

Abstract

Subject Terms

Report Classification unclassified	**Classification of this page** unclassified
Classification of Abstract unclassified	**Limitation of Abstract** UU

Number of Pages
4

A. Sensory models

Proprioceptive sensory models, such as muscle spindle, Golgi tendon organ and Renshaw cells, are being developed for analysis of sensorimotor control functions. The muscle spindle is the most important and complex of the three sensors and a model for it is currently under development. Each spindle provides muscle length and velocity information to the CNS, where limb position is calculated. In turn, the CNS actively modulates the specific sensitivity of the muscle spindle. The model we are developing consists of three blocks representing three types of intrafusal muscle fibers (bag 1, bag 2 and chain fiber). Each intrafusal fiber is modeled as a nonlinear model, which is modified from the lumped linear spindle model suggested by McMahon [3]. Each intrafusal fiber is divided into a central sensory region, which is modeled as a pure elastic element with a sensor whose firing is linearly related to strain, and a polar region, which is modeled as a spring with a parallel contractile element. The contractile element consists of an active force generator, whose force is proportional to static fusimotor input plus baseline force, and a damping element, whose force is proportional to dynamic fusimotor input and nonlinear velocity term ($velocity^{0.3}$) plus a baseline viscosity.

Three separate blocks corresponding to three intrafusal fibers have been constructed in Simulink, as shown in Fig. 1.

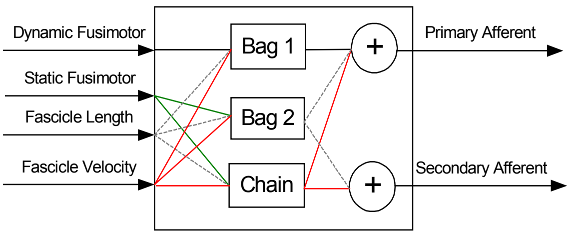

Figure 1. *Simulink model of a generic muscle spindle*

The spindle primary and secondary outputs are combinations of the sensory zone outputs of the intrafusal fibers. The model has 23 parameters (6 for each intrafusal fiber and 5 parameters determining the contribution of fiber outputs to the primary and secondary outputs) that are optimized to reproduce the afferent activity patterns recorded during acute experiments that controlled spindle motion and stimulation of identified fusimotor inputs. The complete spindle block is driven by dynamic and static fusimotor inputs (which must be provided by the neural controller) and the muscle fascicle length and velocity (provided by the muscle block); these are all normalized variables. Because of its normalized inputs and outputs, the spindle model can be easily connected to other NMS model components.

B. Muscle dynamic models

Virtual Muscle is a Matlab-based tool that provides a simple graphical user interface for creating Simulink blocks of muscle. It is composed of two programs, BuildFiberTypes and BuildMuscles, that are used for creating, viewing and manipulating fiber-type and muscle databases. Virtual Muscle comes bundled with sample fiber-type and muscle databases.

The particular muscle model used by Virtual Muscle was based upon an extensive experimental data set collected from feline muscle [4;5]. It provides a more accurate description of muscle force production than any previous model, accounting for the interactive effects of length, velocity and activation over the physiological ranges of each. The model is based upon the premise of modeling specific fiber-types and then summing the effects of different populations of fiber-types to create a whole muscle. The feline fiber-type models have been scaled to human fiber-type models according to published data [6].

The creation of a Simulink muscle block begins with creating models of the fiber-types to be incorporated in the muscle. Users can either use the fiber-types supplied with the basic package or use simple scaling tools in BuildFiberTypes to modify the supplied fiber-types to be either faster or slower. The user can also manipulate every parameter of a fiber-type model if desired. The second step is to describe a muscle in terms of its geometry, number of motor units and fiber-type description using the BuildMuscles function. Once a muscle has been defined, the BuildMuscles function can output a pre-made Simulink Muscle block describing the muscle.

The Simulink muscle block created by BuildMuscles can be incorporated into a user's own Simulink model of the NMS system, or can be incorporated with MMS and SIMM. The input of a typical muscle block consists of an 'activation' and the whole musculotendon length. The output is typically muscle force. However, the user can have other outputs created automatically too, such as fascicle length and velocity, which may be needed by other components such as sensors or reflexes.

The muscle block itself includes the tendon, the fascicles and the muscle mass between them. The fascicles block is further sub-divided into blocks for each motor unit in the muscle. The main block also includes a recruitment block that converts the single activation input into the appropriate activations for each of the motor units. Several recruitment strategies, such as 'natural' or 'FES', are available.

The entire system has been designed so that while the experienced user has the choice to manipulate almost all parameters, the naïve user can avoid the complexities and create complex, realistic models of various muscles using simple tools and commonly available morphometric data. This approach allows the neurophysiologist interested in motor control to focus on the issues that he/she is interested in.

C. Skeletal dynamic models

To model the dynamics of the skeletal system, we have developed the MMS software package [2]. MMS incorporates SIMM's capabilities for constructing anatomically realistic musculoskeletal models and modeling their kinetics. MMS converts those models into Simulink blocks. MMS itself consists of a set of Matlab scripts and C-files that are added to the normal model building process in SIMM. MMS generates compiled C-code that is wrapped in a Simulink S-function, which permits it to be connected to other Simulink blocks and called during simulations. MMS also circumvents limitations in SIMM such as the inability in run-time to generate dynamically changing muscle excitations~~,~~ and external forces and torques, so that neural control systems with feedback can be analyzed. MMS permits the SIMM muscle force-generating model to be bypassed so that the Simulink block that represents the musculoskeletal system can be interfaced with other force-generating models such as Virtual Muscle.

Interactions between the NMS system and the environment can be modeled by applying any number of forces and torques to any of the bone segments in the NMS model. The magnitude and direction and the application point of these external forces and torques can be modified in run-time within Simulink. It is also possible to prescribe the motion of each degree of freedom (DOF) to constrain its motion. In this case, the user can, in run-time and within Simulink, lock or free the motion of the prescribed DOF or force it to move according to any given motion pattern.

D. Neural control models

One of the main reasons for using Simulink as the simulation environment is its easy access to Matlab's control design toolboxes. There are toolboxes for both classical (e.g. servocontrol) and modern (e.g. neural network and fuzzy logic) control systems, which can be used to investigate different ideas on the neural control of movement. These are outside the scope of this presentation.

III. A SAMPLE APPLICATION

A simplified model of the human right arm with 5 rigid segments, 10 DOF, and 10 muscles has been created to demonstrate the capabilities of the integrated modeling environment. First, the modeler uses SIMM to import a set of input files defining the bone shapes, the joints, and the musculotendon paths to graphically reconstruct the musculoskeletal model (Fig. 2).

MMS processing of the SIMM model generates the Simulink model of the musculoskeletal system (Fig. 3). To process the SIMM model, MMS uses SD/FAST (Symbolic Dynamic Inc., USA) and the user-defined model configuration. In this case, the user wishes to prescribe all six DOF of the right shoulder joint, apply a hinge torque to the right elbow joint, apply a point force to the right hand to simulate a load, use SIMM's default muscle models for the first five muscles and Non-SIMM muscle models (Virtual Muscle; see Fig. 4) for the rest. Accordingly, MMS incorporates the appropriate input ports to receive these inputs from other Simulink blocks.

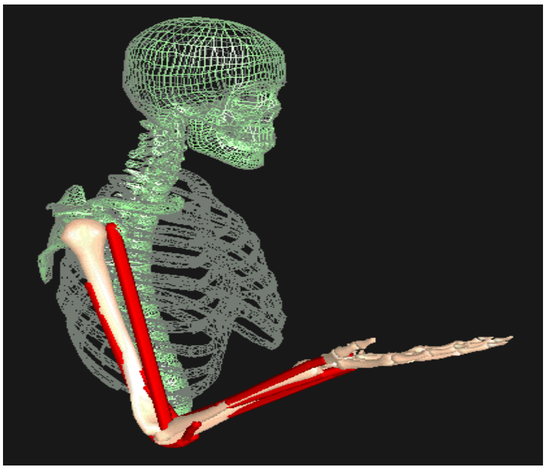

Figure 2. *The anatomical model of the arm in SIMM*

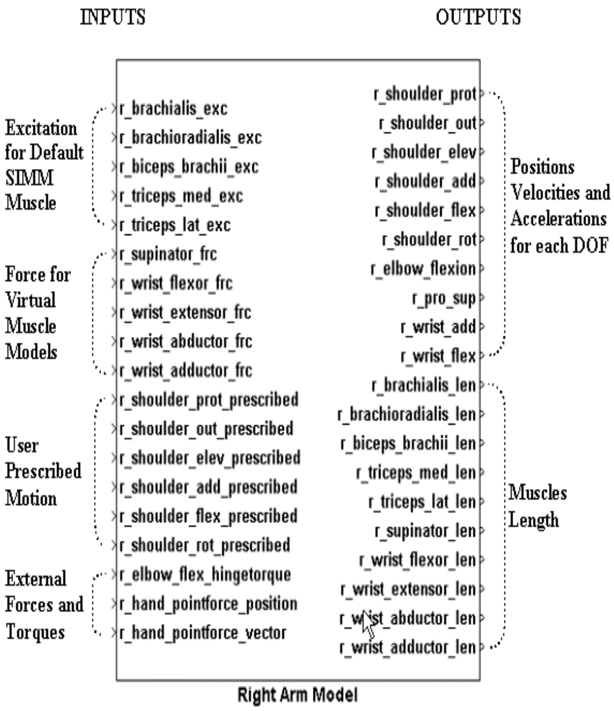

Figure 3. *Simulink model of the arm after MMS*

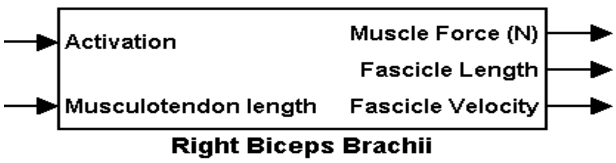

Figure 4*. Simulink model of the right biceps brachii*

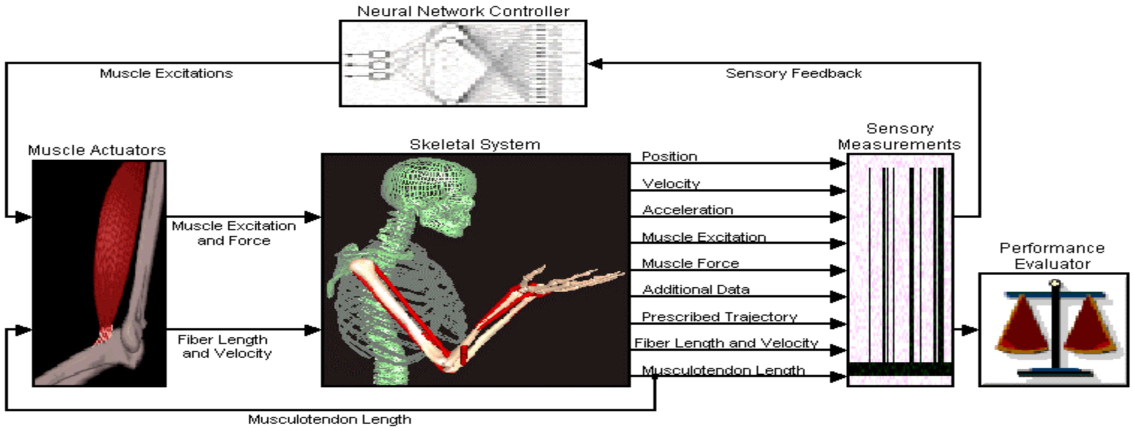

Figure 5. *The complete NMS model of the human right arm in Simulink*

Integration of the above model components with the generic spindle model and other necessary blocks from the Simulink library and Matlab toolboxes completes the NMS model of the right arm (Fig. 5). The model can be provided with a set of input signals and used to generate an output file of joint angle trajectories that can be visualized as an animation of the SIMM model. Multiple simulations from a Matlab script can be used for sensitivity analysis or to optimize a controller for a particular movement.

IV. Discussion

Simulink is a powerful dynamic simulation environment for integration of different components in NMS models. It has many built-in blocks for simulation of typical dynamic systems, which relieves the user from the time-consuming and error-prone process of reprogramming them for a specific project. Matlab's mathematical functions, graphic tools, user interface, and toolboxes are available within Simulink and can be used to enhance NMS models and reduce model development time considerably.

The role of proprioceptive feedback in movement control has been a controversial subject. It is hoped that through modeling and simulation studies, one can gain a coherent understanding of sensorimotor control in NMS systems. The generic spindle model has a physiologically realistic structure, it is relatively simple, and can be easily interfaced with other NMS model components.

Virtual Muscle has been designed so that while the experienced user has the choice to manipulate almost all muscle-specific parameters, the naïve user can avoid the complexities and difficulties of creating and programming their own muscle model, yet still be able to create a complex, realistic model of muscle in a short period of time using simple tools. More importantly, by using a common programming environment such as Simulink, the model can be shared easily and incorporated into larger NMS models easily.

Neural control strategies are the subject of intense research and a variety of control structures have been proposed. Matlab's control toolboxes are very flexible and can be used to model traditional and modern controllers or develop new ones. Their true capabilities and limitations can only be appreciated by linking them with realistic models of the sensorimotor system to be controlled and studying their performance under a wide range of simulated tasks and conditions.

Simulink and Matlab are used widely for teaching and research, which should facilitate sharing and reusing modeled components. Researchers, clinicians and educators in the fields of motor control and biomechanics are among the potential users who may benefit from an integrated modeling environment in which to analyze the control of movement in NMS systems.

References

[1] Delp, S. L. and Loan, J. P., "A computational framework for simulating and analyzing human and animal movement," *Computing in Science & Engineering*, vol. 2, no. 5, pp. 46-55, 2000.

[2] Davoodi, R. and Loeb, G. E., "Conversion of SIMM to SIMULINK for faster development of musculoskeletal models," IFESS 2001.

[3] McMahon, T. A., "Muscles, reflexes, and locomotion," Princeton University Press, pp. 139-167, 1984.

[4] Brown, I. E., Cheng, E. J., and Loeb, G. E., "Measured and modeled properties of mammalian skeletal muscle. II. The effects of stimulus frequency on force-length and force-velocity relationships," *J.Muscle Res.Cell Motil.*, vol. 20, no. 7, pp. 627-643, 1999.

[5] Brown, I. E. and Loeb, G. E., "Measured and modeled properties of mammalian skeletal muscle: IV. dynamics of activation and deactivation," *J.Muscle Res.Cell Motil.*, vol. 21, no. 1, pp. 33-47, 2000.

[6] Cheng, E. J., Brown, I. E., and Loeb, G. E., "Virtual muscle: a computational approach to understanding the effects of muscle properties on motor control," *J.Neurosci.Methods*, vol. 101, no. 2, pp. 117-130, 2000.

Acknowledgment

Supported by the A.E. Mann Institute for Biomedical Engineering